Ticket Show Application

Author:
Name: Anupam Kumar Jha
Roll No.: 21f1004905
Student Email id: 21f1004905@ds.study.iitm.ac.in
About: I am currently in the diploma level of BS program from IIT Madras. I am from Rajsamand, Rajasthan. I am a part of an international organization called Toastmasters which works on developing communication and leadership skills. I like to watch Cricket, teach, and love to do yoga and meditation.

Description:
In the ticket show application a user with admin access can create new theatres and shows as Ill as update and delete the existing theatres and shows. The admin also can import the theatre data in csv format. A normal user can search the theatre with the location and search the show with name and book the tickets. The user also can see his profile.. The app will automatically send reminder mail to the user who hasn't booked any ticket for a long time. And the app also sends a monthly report to the user in his/her mail.

Technologies Used:

1. **Frontend**: In the frontend, I used three languages: HTML, CSS, JavaScript. I have also used the **vue** framework to implement the reactivity on the app.
2. **Backend**: Following libraries Ire used to implement the different parts of the backend.
    1) Flask
    2) Jinja2
    3) Flask-SQLAlchemy==3.0.3
    4) Flask-security-too
    5) Email_validator
    6) Bcrypt
    7) Flask_restful
    8) Flask-Cors
    9) Pytz
    10) Redis
    11) Celery[redis]
    12) Flask-SSE
    13) lasyprint=52.5
    14) Flask-Caching
3. **Data Storage**: The database that is used is SQLite. And I used DB Browser for manipulating the database graphically.
4. **Others:** I have used **VS-Code** for writing the codes, **Ubuntu Terminal** to execute the codes, and **Chrome** for testing purpose.

Architecture: I have used a **JAM** approach that is JavaScript, API, Markup. I separated the architecture in two different segments. In which all the data manipulation happens through javascript and frontend gets all the data through API from backend. And Markup is used to render those manipulative data that forms our application's view.

Frontend: To develop the frontend part of the application I use **JavaScript, HTML and CSS**. I have used JavaScript's **Vue** framework to implement the reactivity into the application. I also used **Bootstrap** for styling purposes.

Backend API: I have created CRUD APIs for data transfer from backend to frontend. I used the **flask-restful** library for implementing the API. I have implemented APIs for creating, updating and deleting and getting the details of theatres and shows. There are some other APIs that have also been implemented like Profile API, Booking API, ExportDetails API.

Backend Jobs: Some backend jobs have been implemented like sending daily reminder mail, monthly report and exporting theatres details using **celery and redis-server**.

Features:

1. ***Core***:
   (i) *User Signup and Login*:- With the flask-security-too I have implemented login for admin as well as for normal users.
   (ii) *Theatre Management*: The admin can manage theatres. He/She can create a new theatre as well as update and delete the existing theatre.
   (iii) *Show Management*: The admin can manage shows. He/She can create a new show as well as update and delete the existing show and allocate theatre to shows.
   (iv) *Search for shows/theatres*: The user can search the shows based on show name and search theatres based on location.
   (v) *Book Show Tickets*: The user can book the tickets of the shows(If it's not housefull).
   (vi) Daily Reminder Job: A user will get a daily email if he/she hasn't booked a ticket on a particular day.
   (vii) Monthly Entertainment Report: A user will get a monthly report on mail on every 1st day of the month of their activities on the site.
   (viii) Export as CSV: A admin can export theatre details like no. of shows, their time, date, price, available seats and price in csv format.
   (ix) Performance and Caching: To increase the performance and efficiency of the app I also implemented caching where there was need.

2. ***Recommended:***
   (i) Well Design PDF: I have implemented a function to create a well-designed pdf for the user's monthly report.
   (ii) Single Responsive UI: The implemented UI can work across all the devices.

Video-Link:
https://drive.google.com/file/d/1UM_ygtrLt2idyrRJDrk_AAMprxCwTrc_/view?usp=sharing